PostgreSQL Replicator – easy way to build a distributed Postgres database

Irina Sourikova

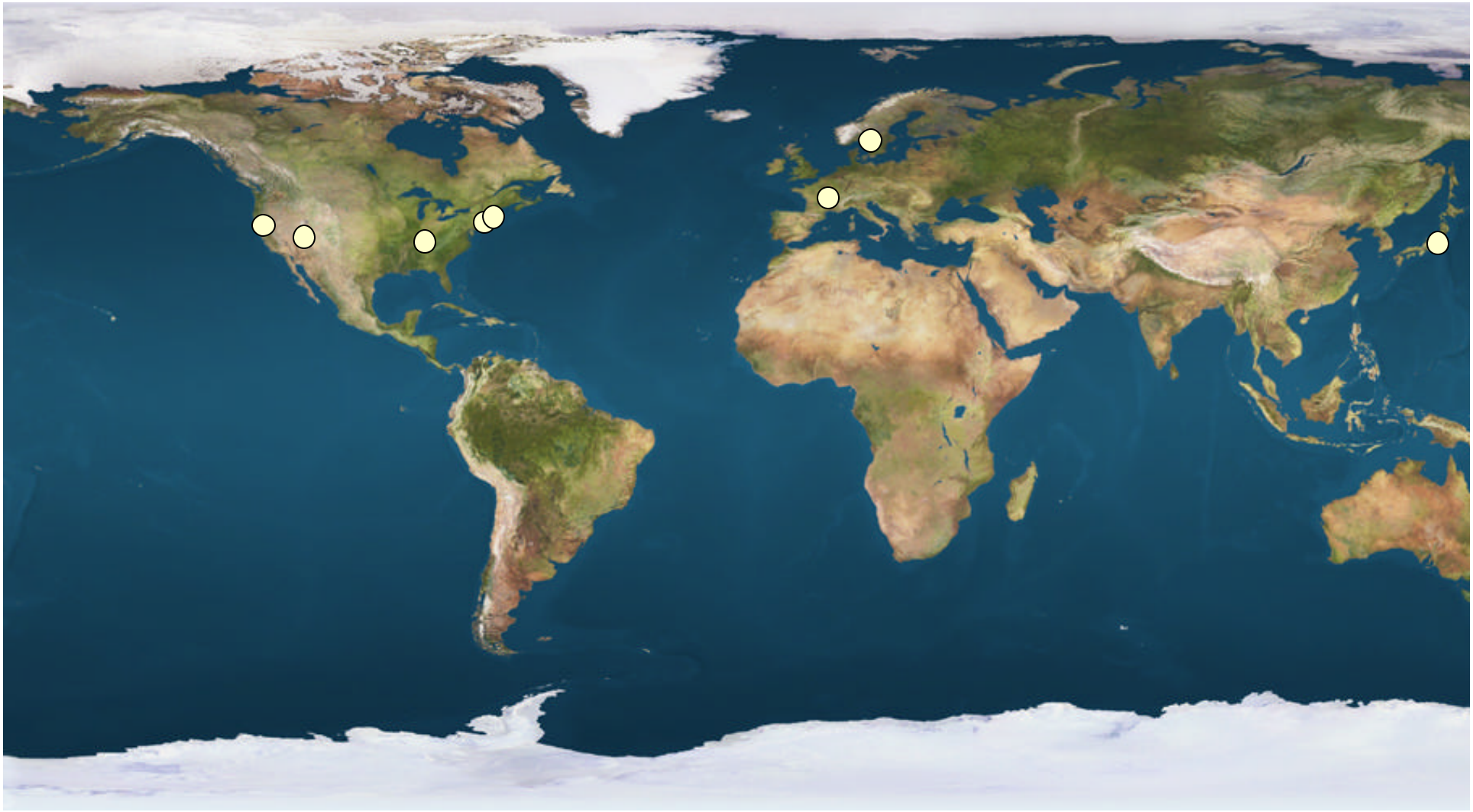
PHENIX collaboration

Outline

- Why PHENIX had to search for another DB technology
- Options: Objectivity, Oracle, MySQL, Postgres
- Postgres DB
- Postgres Replicator
- Distributed PHENIX File Catalog
- File Catalog API: FROG
- Conclusions

PHENIX

Major PHENIX Computing Sites



BNL

LLNL

SUNYSB

Lund

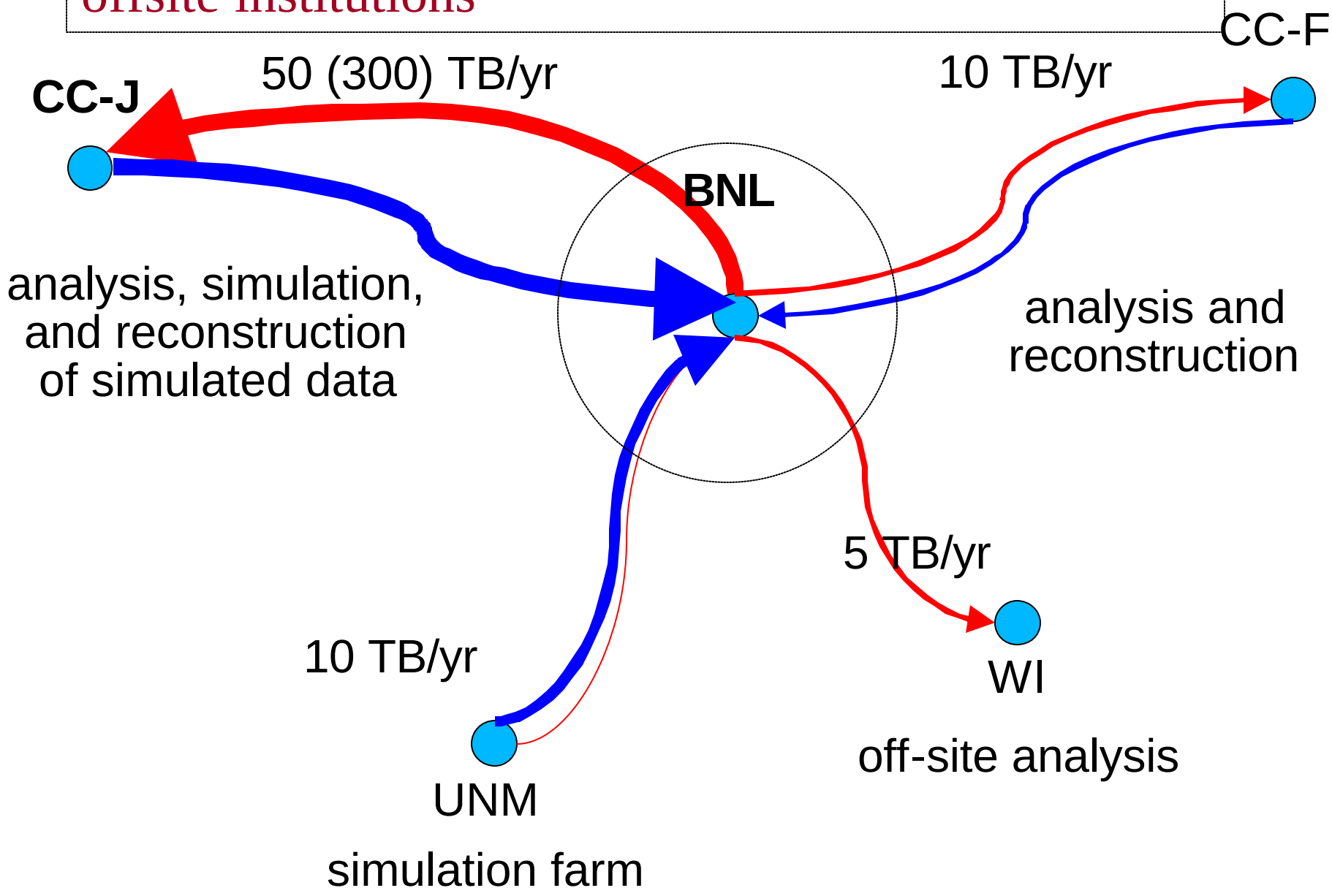
UNM

VU

CC-F

CC-J

Large amount of data is moved between BNL and offsite institutions



Central File Catalog doesn't scale in distributed environment

- Because of WAN latency central File Catalog was not updated online during production at remote sites
- Remote sites maintained their own catalogs of local files

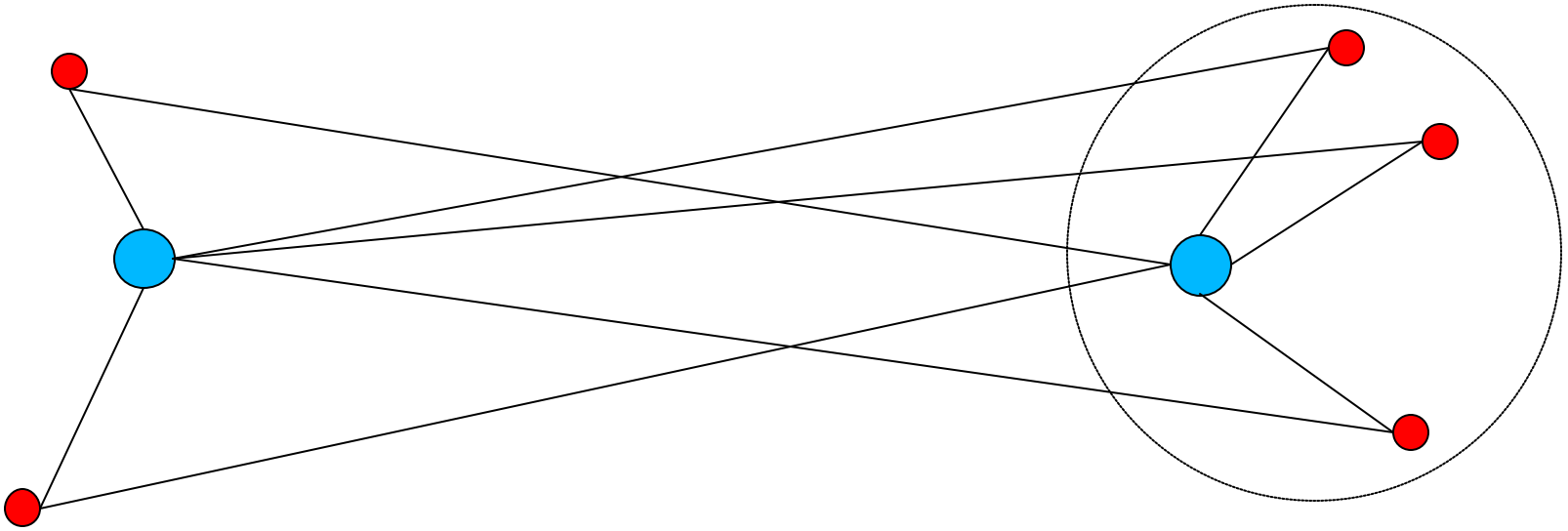
Was hard to keep File Catalogs up-to-date, required a lot of work

PHENIX

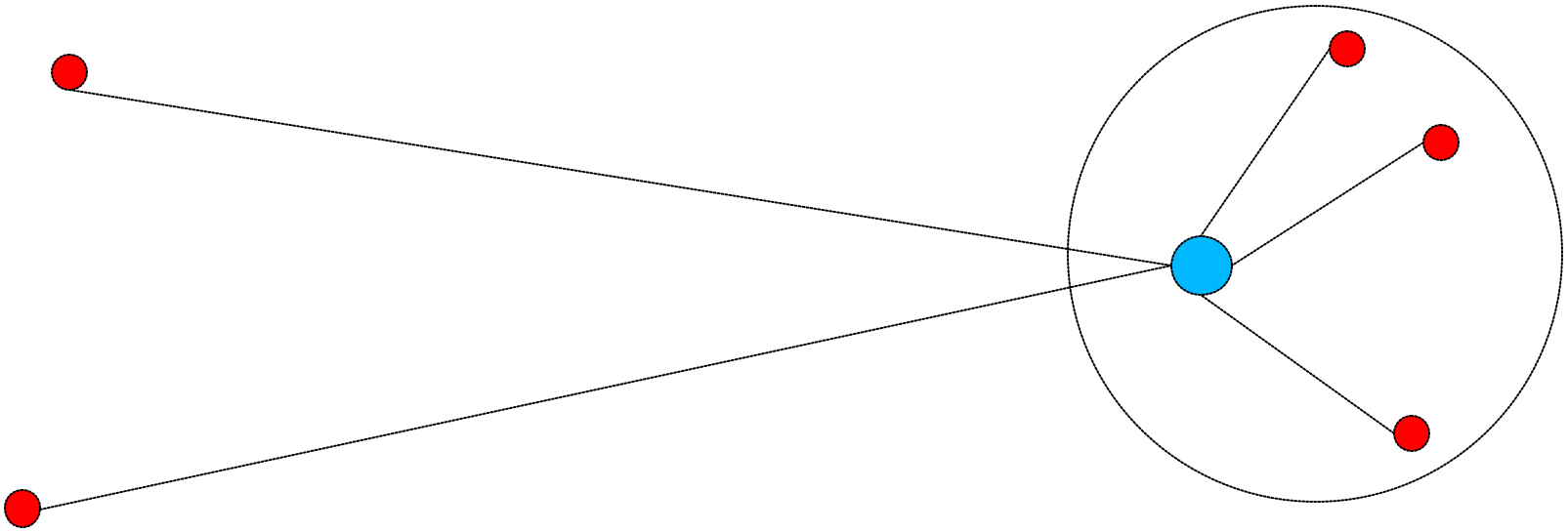
What we were looking for

- Provide write permissions to the sites that store portions of the data set

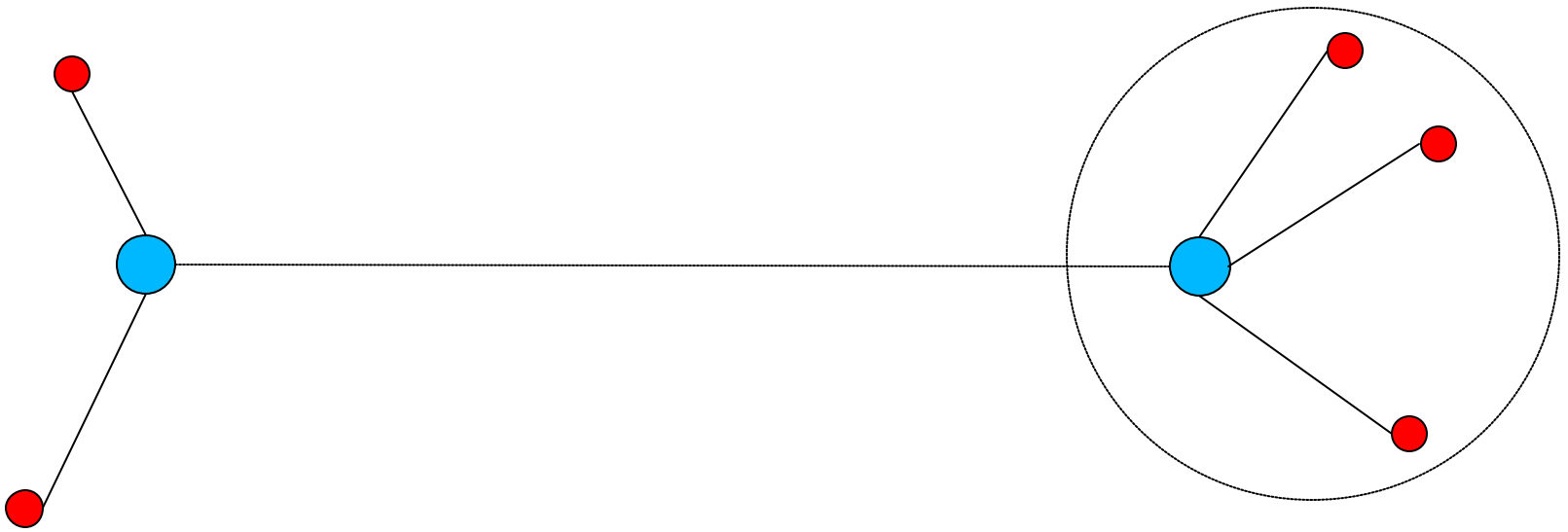
PHENIX



Objy: difficulty of PDN negotiation, exposes primary DB to world



WAN latency, single-point-of-failure



What we needed: hybrid solution

Database technology choice

- Objectivity – problems with peer-to-peer replication
- Oracle was an obvious candidate(but expensive)
- MySQL had only master-slave replication
- PostgreSQL seemed a very attractive DBMS with several existing projects on peer-to-peer replication
- SOLUTION: to have central Objy based metadata catalog and distributed file replica catalog

PHENIX

PostgreSQL

- ACID compliant
- Multi-Version Concurrency Control for concurrent applications
- Easy to use
- Free
- LISTEN and NOTIFY support message passing and client notification of an event in the database. Important for automating data replication

PHOENIX

PostgreSQL Replicator

http://pgreplicator.sourceforge.net

- Works on top of Postgres DB
- Partial, peer-to-peer, async replication
- Table level data ownership model
- Table level replicated database set

PHOENIX

Distributed PHENIX Catalog

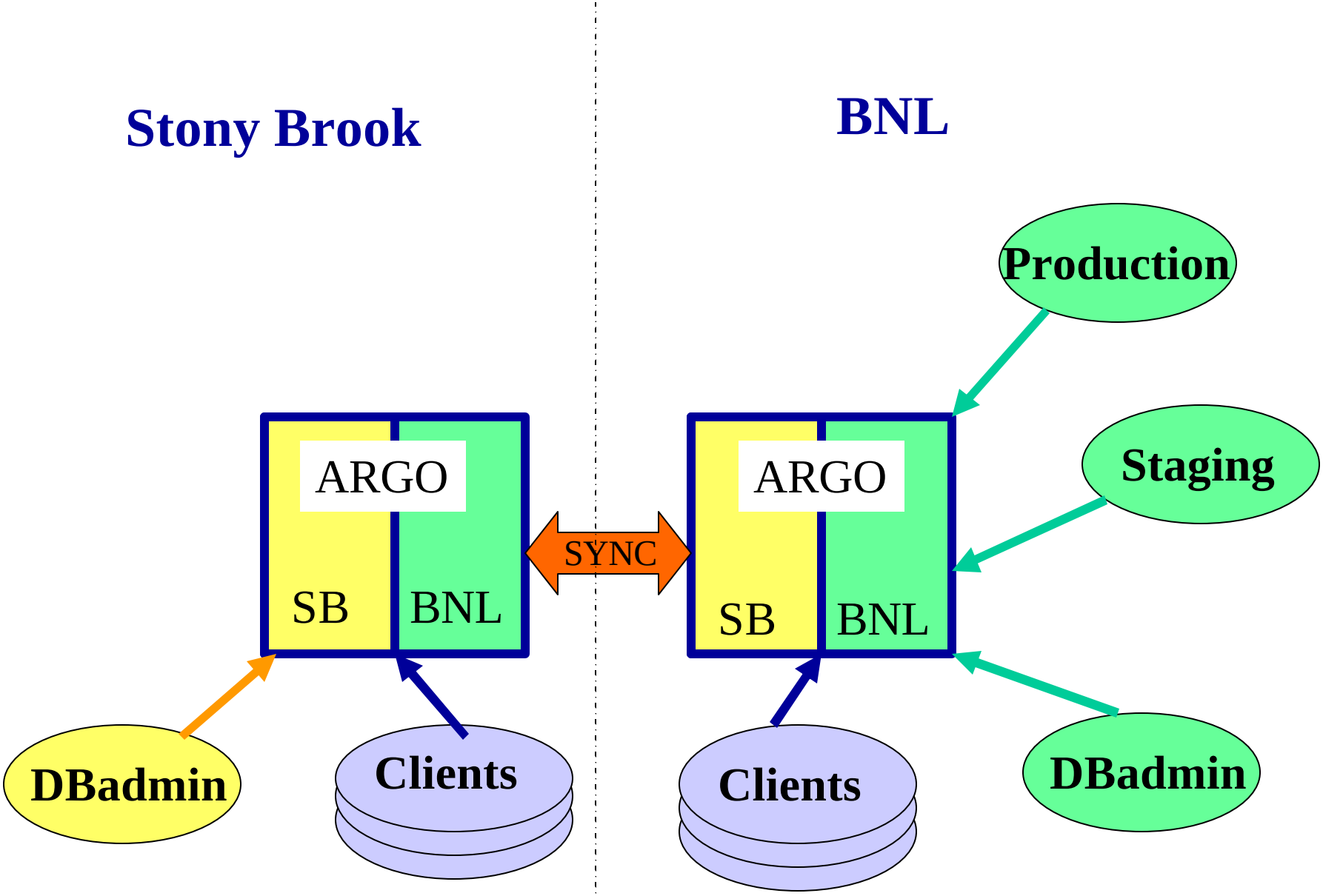
- Distribution includes BNL and Stony Brook University
- Two sites have common tables that are synchronized during replica session
- Common tables have Workload Partitioning data ownership model, that means the table entries can be updated only by the site-originator

PHENIX

Table ‘files’

LFN	host	filePath	fsize	time
xyz.dst	XXX.rcf.bnl.gov	/phenix/data01/dsts/xyz.dst	12345678	2003-3-8 11:20:03
EVENTDATAXXX.prdf	HPSS	/phenix/phnxsink/…		
123.root	ram3.chem.sunysb.edu	/…./123.root		
123.root	ZZZ.rcf.bnl.gov	/phenix/data45/…/123.root	22334455	2003-1-1 09:00:02

PHENIX


Stony Brook
BNL
Production
Staging
ARGO
ARGO
SYNC
SB
BNL
SB
BNL
DBadmin
Clients
Clients
DBadmin

File Catalog replication

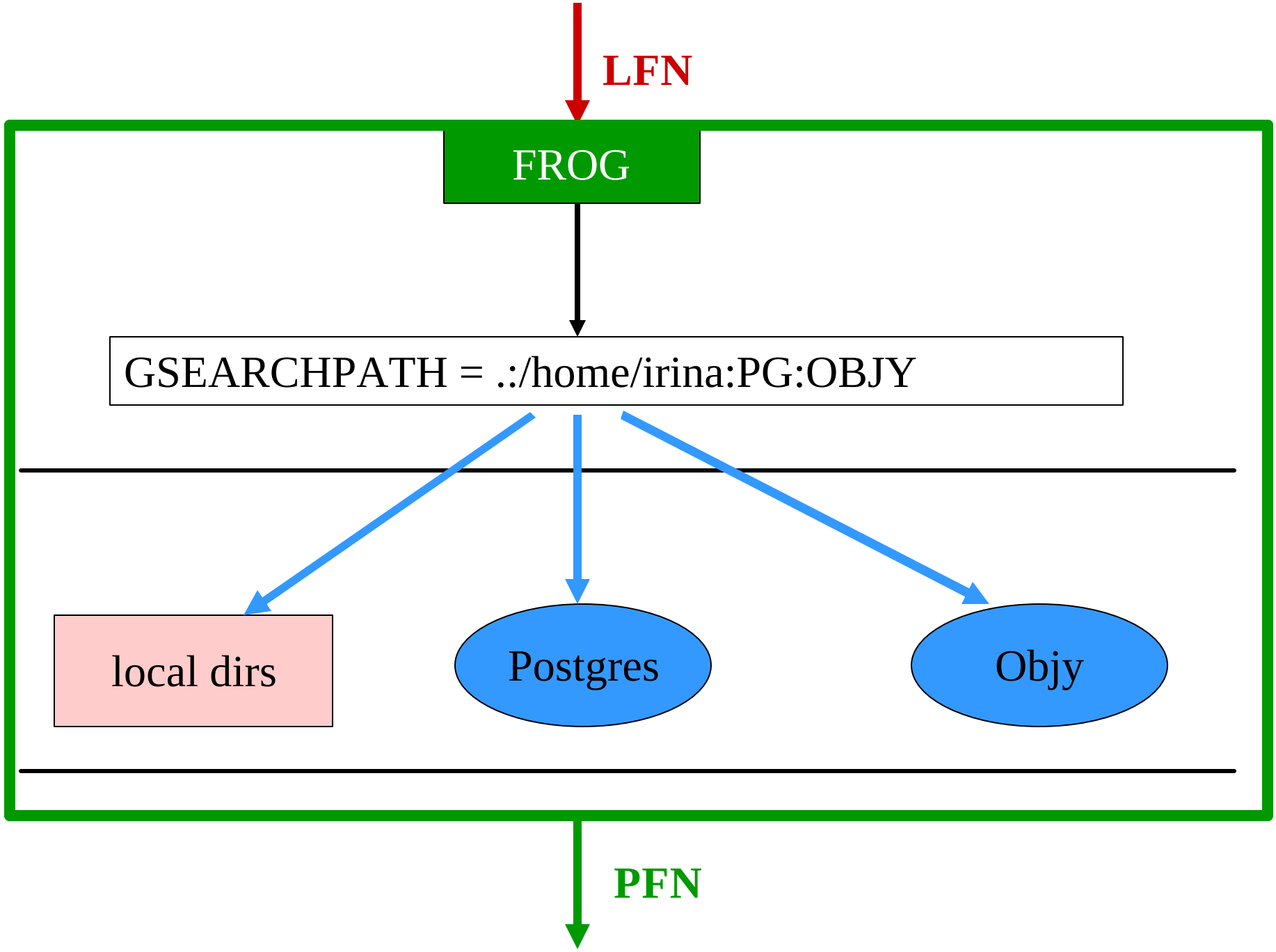
- Synchronization is two-way
- Fault tolerant
- DB replication is partial, only latest updates are transferred over the network
- That solves scalability problems
- Replication of 20 000 new updates takes 1min

PHENIX

File Catalog API: FROG

- File Replica On Grid
- Translates Logical File Name to Physical File Name
- Shields users from Database implementation
- User code stays the same in different environment – with or without DB access(important for offsite analysis)

PHENIX


LFN
FROG
GSEARCHPATH = .:/home/irina:PG:OBJY
local dirs
Postgres
Objy
PFN

‘Least surprise’ design principle

- FROG has 1 method -
  `const char * location(const char * lfn)`
- If lfn starts with a / , it is considered as an absolute file path and FROG::location() returns it
- If env GSEARCHPATH is not set, FROG::location() returns lfn
- If users don’t have DB access, env GSEARCHPATH can be set accordingly

PHENIX

Plug-in architecture

- To search different databases FROG loads the corresponding dynamic libraries
- Easy to change database backend(user code not affected)

PHENIX

What do we gain by using FROG?

- No hardwired file paths in the code
- Analysis code still works when files are relocated
- Possible to replicate very popular files
- Extremely useful for NFS load balancing
- Better use of available resources
- Faster analysis

Conclusions

- Postgres Replicator is a free software that allows to build a distributed database
- This technology allowed PHENIX to distribute File Catalog and have the following advantages over central FC:
  - Reliability(no single point of failure)
  - Accessibility(reduced workload, no WAN latency)
  - Solved security problems (firewall conduits only between servers)
  - Catalog data is more up-to-date(all sites update the catalog online)
  - Less manpower needed to maintain the catalog

PHENIX